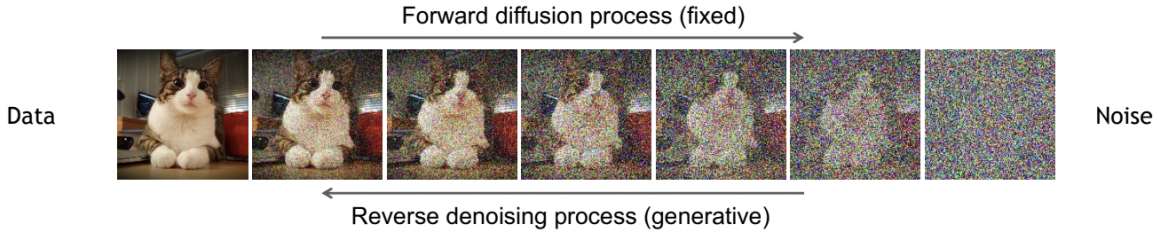
Classifier Guidance on Diffusion Models

Introduction

The paper “Diffusion-LM Improves Controllable Text Generation ” develops a non-autoregressive language model based on *continuous* diffusions. The diffusion based language model iteratively denoises a sequence of Gaussian vectors into word vectors, yielding a sequence of intermediate latent variables. These intermediate variables enables a simple gradient-based algorithm to perform complex,controllable generation tasks. The gradients act as a guide to move the diffusion sampling process towards an arbitrary class label.

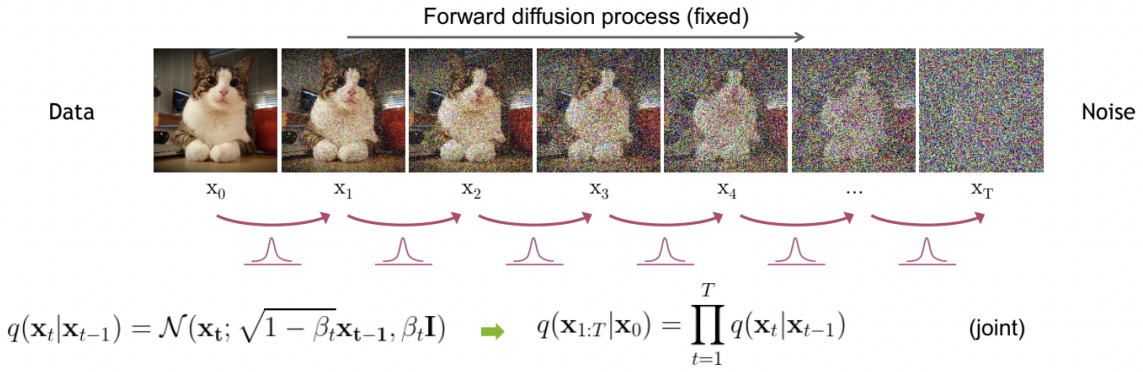
Denoising Diffusion Probabilistic model

Diffusion models are generative models and are inspired by non equilibrium thermodynamics. They define a markov chain of diffusion steps where they learn to destroy or corrupt training data by slowly adding random noise to it and then learn to recover the data by reversing this noising process. The latent variables dimensionality is same as the original data and does not reduce unlike the other generative models.Diffusion models consist of two process : a predefined deterministic forward diffusion and a learned reverse denoising diffusion process.



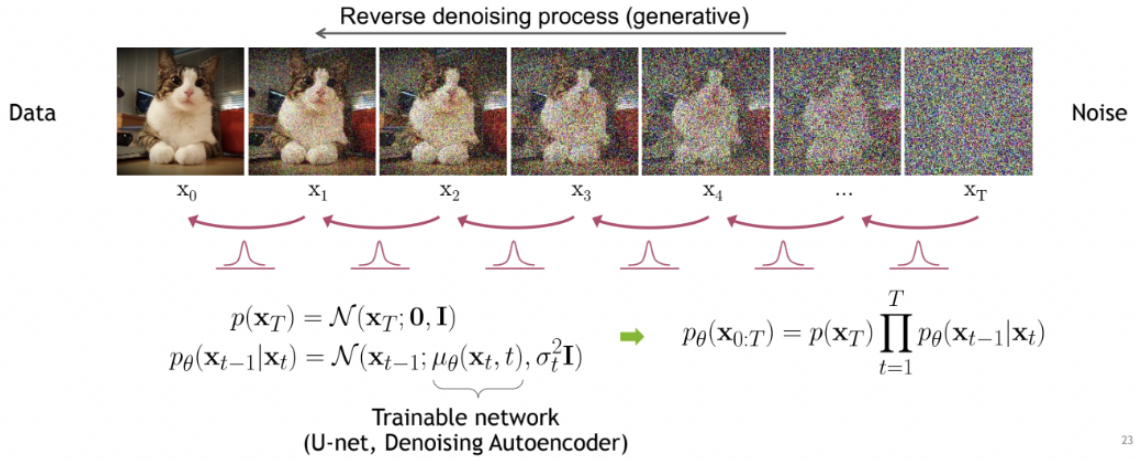
Forward Diffusion Process

In the forward diffusion process we sample a data point $\mathbf{x_0}$ from real data distribution **q(x) .** To this we incrementally add small amount of gaussian noise to data $\mathbf{x_0}$ until at time step **T** , samples $\mathbf{x_T}$ are approximately gaussian. Each transition $\mathbf{x_{t-1}} \mathbf{—> x_t}$ is parameterised as shown below where the hyperparameter **ß** is the amount of noise added at each time step **t**. The forward process contains no trainable parameters and allows us to define a training objective that involves generating noisy data according to a pre-defined forward process **q** and training a model to reverse the process and reconstruct the data.



The Reverse Diffusion Process

Diffusion models sample from a distribution by reversing a gradual noising process. In particular,sampling starts with noise $\mathbf{x_T} \sim \mathbf{N(0,I)}$ and produces gradually-less-noisy samples $\mathbf{x_{T-1}}$,$\mathbf{x_{T-2}}$,... until reaching a final sample $\mathbf{x_0}$. Each timestep t corresponds to a certain noise level, and x_t can be thought of as a mixture of a signal $\mathbf{x_0}$ with some noise $\boldsymbol{\varepsilon}$ where the signal to noise ratio is determined by the timestep **t**. However we cannot easily estimate $\mathbf{q(x_{t-1} \mid x_t)}$ therefore we need to learn a model $\mathbf{p_\theta}$ to approximate the conditional probability in the reverse diffusion process where the mean can be calculated as a function of $\boldsymbol{\varepsilon_\theta(x_t, t)}$



The diffusion model is trained to maximize the marginal likelihood of the data $\mathbf{E}_{x0 \sim p_{data}}[\log p_\theta(x_0)]$, and the canonical objective is the variational lower bound of $\log p_\theta(x_0)$

$$\mathcal{L}_{\text{vlb}}(\mathbf{x}_0) = \mathop{\mathbb{E}}_{q(\mathbf{x}_{1:T}|\mathbf{x}_0)} \left[\log \frac{q(\mathbf{x}_T|\mathbf{x}_0)}{p_\theta(\mathbf{x}_T)} + \sum_{t=2}^{T} \log \frac{q(\mathbf{x}_{t-1}|\mathbf{x}_0, \mathbf{x}_t)}{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)} - \log p_\theta(\mathbf{x}_0|\mathbf{x}_1) \right]. \quad (1)$$

This objective function can be unstable and require many optimisation tricks to stabilise. Therefore the authors devised a simple surrogate objective that expands and reweights each KL-divergence term in L_{vlb} to obtain a mean-squared error loss which will be referred to as :

$$\mathcal{L}_{\text{simple}}(\mathbf{x}_0) = \sum_{t=1}^{T} \mathop{\mathbb{E}}_{q(\mathbf{x}_t|\mathbf{x}_0)} ||\mu_\theta(\mathbf{x}_t, t) - \hat{\mu}(\mathbf{x}_t, \mathbf{x}_0)||^2,$$

where $\hat{\mu}(x_t, x_0)$ is the mean of the posterior $q(x_{t-1}|x_0, x_t)$ which is a closed from Gaussian, and $\mu_\theta(x_t, t)$ is the predicted mean of $p_\theta(x_{t-1} | x_t)$ computed by a neural network. While **Lsimple** is no longer a valid lower bound, prior work has found that it empirically made training more stable and improved sample quality .

Classifier Guided Diffusion

Here a pre-trained diffusion model can be conditioned using the gradients of a classifier. In particular, can train a classifier $p_\phi(y|x_t, t)$ on noisy images x_t, and then use gradients $\nabla_{x_t} \log p_\phi(y|x_t, t)$ to guide the diffusion sampling process towards an arbitrary class label y.
To explain by example , suppose you want to sample only dog images then the mean of the reverse process is shifted in the direction towards the sub space having dog images.

Algorithm 1 Classifier guided diffusion sampling, given a diffusion model $(\mu_\theta(x_t), \Sigma_\theta(x_t))$, classifier $p_\phi(y|x_t)$, and gradient scale s.

Input: class label y, gradient scale s
$x_T \leftarrow$ sample from $\mathcal{N}(0, \mathbf{I})$
for all t from T to 1 **do**
  $\mu, \Sigma \leftarrow \mu_\theta(x_t), \Sigma_\theta(x_t)$
  $x_{t-1} \leftarrow$ sample from $\mathcal{N}(\mu + s\Sigma \nabla_{x_t} \log p_\phi(y|x_t), \Sigma)$
end for
return x_0

We have thus found that the conditional transition operator can be approximated by a Gaussian similar to the unconditional transition operator, but with its mean shifted by Σg.

Diffusion model from scratch : colab inspired from Diffusion models from scratch in PyTorch
Clip guided Diffusion model : colab inspired from link